Nonparametric regression

Examples of nonparametric regression

The first examples of nonparametric regression are the familiar scatter diagram smoother `lowess()` and the related, more flexible `loess()` function.

Load some packages, and attach a data set of annual temperatures at Oregon climate stations.

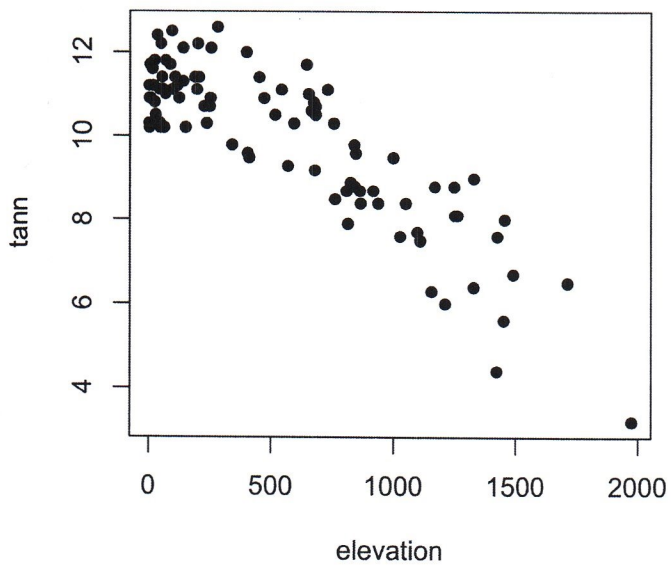
```
library(RColorBrewer)
library(sf)
```

```
attach(ortann)
names(ortann)
```

```
## [1] "station"   "latitude"  "longitude" "elevation" "tann"
```

Look at the annual temperature data:

```
plot(elevation, tann, pch=16)
```



Note that there are actually two versions of the lowess or loess scatter-diagram smoothing approach implemented in R `lowess()` and `loess()` . The former function (`lowess()`) was implemented first, while the latter (`loess()`) is more flexible and powerful. Because the function names are pronouced similarly, they are often confused, but because they basically do the same thing, that's not such a big deal.

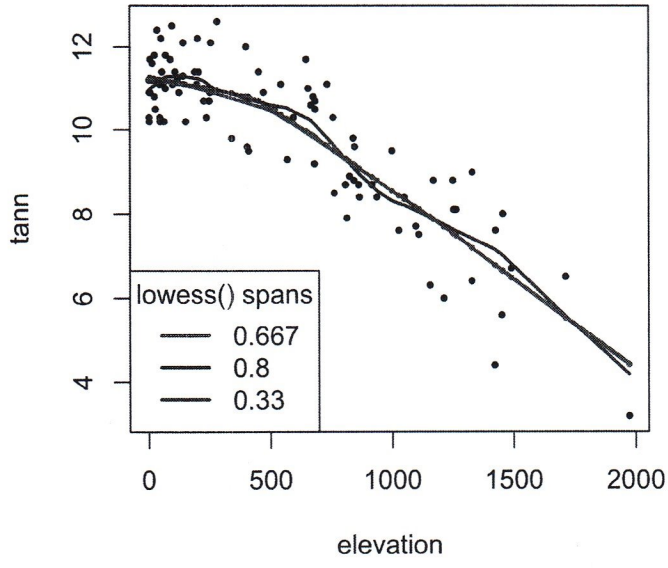
3.1 lowess()

A simple "lowess/loess" curve is constructed using the `lowess()` function, which finds a "fitted" value for each data point; these can be plotted as individual symbols, but they are usually connected with lines. The lowess() function has a "span" argument (sometimes symbolized by λ) that represents the proportion of the total number of points that contribute to each local fitted value. In practice, the `lowess()` function is often embedded in a `points()` or 'lines()' function.

In the following, the specific fitted values produced by the `lowess()` function (one per data point) are plotted in red, and the lowess curve is added, also in red. On top of that, two additional curves are plotted (in magenta and purple) that show the effect of the smoothing parameter choice. The green curve is smoother than the default ($\lambda = 0.667$), and the magenta curve is "looser" than the default ($\lambda = 0.33$).

```
# lowess
plot(elevation, tann, pch=16, cex=0.6)
points(lowess(elevation, tann), pch=16, col="red", cex=0.5)
lines(lowess(elevation,tann), col="red", lwd=2)

# different smoothing
lines(lowess(elevation, tann, f=0.33), col="blue", lwd=2)
lines(lowess(elevation, tann, f=0.80), col="purple", lwd=2)
legend("bottomleft", title = "lowess() spans", legend=c("0.667","0.8","0.33"), lwd=2, cex=1, col=c("red","blue","purple"))
```



3.2 loess()

The newer **`loess()`** function uses a formula to specify the response (and in its application as a scatter-diagram smoother) a single predictor variable. The **`loess()`** function creates an object that contains the results, and the **`predict()`** function retrieves the fitted values. These can then be plotted along with the response variable. However, the points must be plotted in increasing order of the predictor variable values in order for the **`lines()`** function to draw the line in an appropriate fashion. This is done by using the results of the **`order()`** function applied to the predictor variable values, and the explicit subscripting (in square brackets **`[ ]`**) to arrange the observations in ascending order.

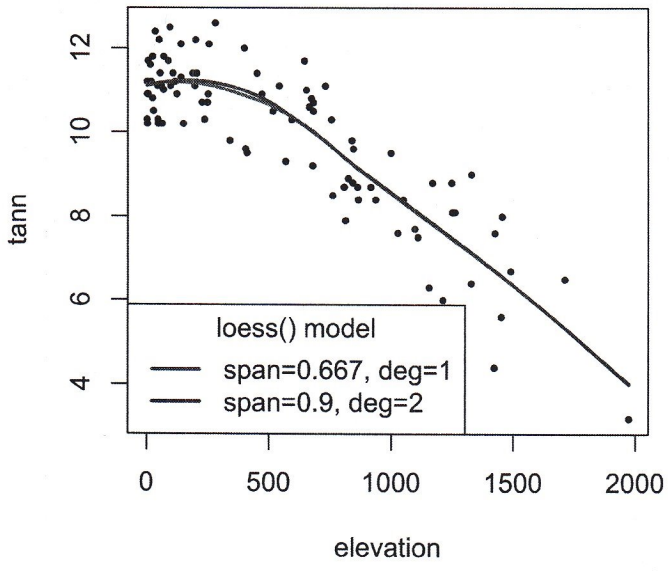
```
# loess -- first model
loess_model <- loess(tann ~ elevation)
loess_model
```

```
## Call:
## loess(formula = tann ~ elevation)
##
## Number of Observations: 92
## Equivalent Number of Parameters: 4.69
## Residual Standard Error: 0.8758
```

```
# second model, smoother curve
loess_model2 <- loess(tann ~ elevation, span=0.90, degree=2)
loess_model2
```

```
## Call:
## loess(formula = tann ~ elevation, span = 0.9, degree = 2)
##
## Number of Observations: 92
## Equivalent Number of Parameters: 4.04
## Residual Standard Error: 0.8752
```

```
# plot the curves
plot(tann ~ elevation, pch=16, cex=0.6)
hat1 <- predict(loess_model)
lines(elevation[order(elevation)], hat1[order(elevation)], col="red", lwd=2)
hat2 <- predict(loess_model2)
lines(elevation[order(elevation)], hat2[order(elevation)], col="blue", lwd=2)
legend("bottomleft", title = "loess() model", legend=c("span=0.667, deg=1","span=0.9, deg=2"), lwd=2, cex=1, col=c("red","blue"))
```



3.3 loess() surfaces

A locally determined surface can be constructed using **`loess()`** which is not limited to a single predictor variable, by first fitting a model that illustrates the response of the response variable as a function of two (or more) location variables, and then using the **`predict()`** function to visualize the resulting surface:

```
# tann as a function of latitude and longitude (and interaction)
tann_loess <- loess(tann ~ longitude + latitude, span=0.3)
summary(tann_loess)
```

```
## Call:
## loess(formula = tann ~ longitude + latitude, span = 0.3)
##
## Number of Observations: 92
## Equivalent Number of Parameters: 21.11
## Residual Standard Error: 0.9942
## Trace of smoother matrix: 25.23  (exact)
##
## Control settings:
##   span     :  0.3
##   degree   :  2
##   family   :  gaussian
##   surface  :  interpolate      cell = 0.2
##   normalize:  TRUE
##  parametric:  FALSE FALSE
## drop.square:  FALSE FALSE
```

```
# poor man's R-squared value
cor(tann, tann_loess$fitted)^2
```

```
## [1] 0.8098961
```

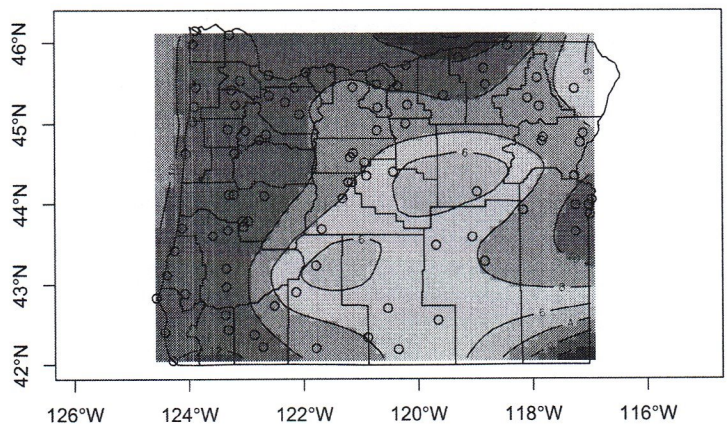
Now create an interpolation "target" grid, get the predicted (i.e. interpolated) values, and plot the results.

```
# create an interpolation target grid to display predicted values
grid_longitude <- seq(-124.5000, -116.8333, .1667)
grid_latitude <- seq(42.0000, 46.1667, .0833)
grid_mar <- list(longitude=grid_longitude, latitude=grid_latitude)

# get the fitted (interpolated) values
tann_interp <- predict(tann_loess, expand.grid(grid_mar))
tann_z <- matrix(tann_interp, length(grid_longitude),
length(grid_latitude))

# plot the interpolated values as shaded rectangles and contours
nclr <- 8
plotclr <- brewer.pal(nclr, "PuOr")
plotclr <- plotclr[nclr:1] # reorder colors

plot(st_geometry(orotl_sf), axes=TRUE)
image(grid_longitude, grid_latitude, tann_z, col=plotclr, add=T)
contour(grid_longitude, grid_latitude, tann_z, add=TRUE)
points(longitude, latitude)
plot(st_geometry(orotl_sf), add=T)
```



3.4 Other bivariate smoothers

Loess is one of a number of smoothers (including linear regression as an end-member) that can be used. The different smoothers vary in the assumptions they make about

- the form of the relationship
- the influence of individual points

The other scatter diagram smoothers include a straight, or "least-squares" line, a low-order polynomial least-squares line, and the "smoothing spline". Each can be viewed as special cases of the more flexible loess-type smoothers in which the curve is very simple. The best way to understand these different smoothers is to compare them:

3.4.1 Least-squares line

```
# first-order polynomial (i.e. a straight line)
linear_model <- lm(tann ~ elevation)
linear_model
```

```
##
## Call:
## lm(formula = tann ~ elevation)
##
## Coefficients:
## (Intercept)    elevation
##   11.688139    -0.003238
```

3.4.2 Least-squares polynomials

```
# second order polynomial
poly2_model <- lm(tann ~ elevation+ I(elevation^2))
poly2_model
```

```
##
## Call:
## lm(formula = tann ~ elevation + I(elevation^2))
##
## Coefficients:
##    (Intercept)       elevation  I(elevation^2)
##      1.133e+01      -1.147e-03      -1.459e-06
```

```
poly2_hat <- predict(poly2_model)
```

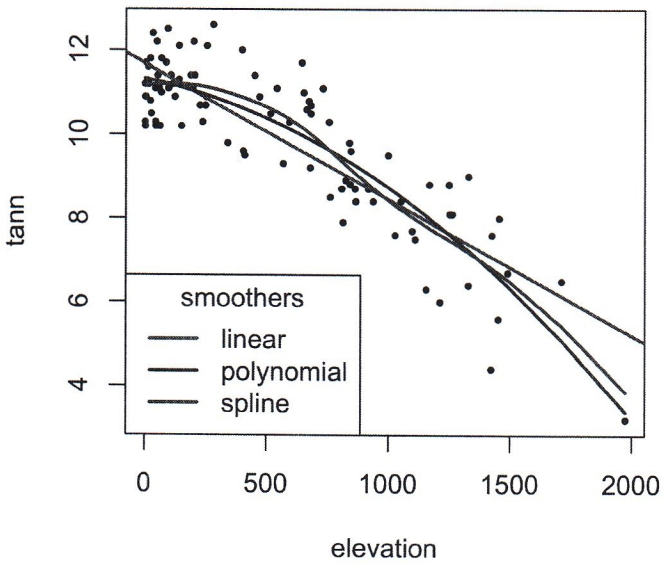
3.4.3 Smoothing spline

```
spline_model <- smooth.spline(elevation, tann)
spline_model
```

```
## Call:
## smooth.spline(x = elevation, y = tann)
##
## Smoothing Parameter  spar= 1.12444  lambda= 0.003202393 (15 iterations)
## Equivalent Degrees of Freedom (Df): 5.311004
## Penalized Criterion (RSS): 63.56046
## GCV: 0.800279
```

Plot the differnt smoothers

```
plot(tann ~ elevation, pch=16, cex=0.6)
abline(linear_model, col="red", lwd=2)
lines(elevation[order(elevation)], poly2_hat[order(elevation)],
     col="blue", lwd=2)
lines(spline_model, col="purple", lwd=2)
legend("bottomleft", title = "smoothers", legend=c("linear","polynomial","spline"), lwd=2, cex=1, col=c("red","blue","purple"))
```



[Back to top]

4 Summary of smoothers

The various smoothers can be summarized as follows

Assumptions	Smoother	Form	Influence of individual points
fewest	loess	no assumptions	unusual points discounted
	smoothing spline	smooth curve	some discounting of unusual points
	robust, robust MM	straight line	unusual points discounted
	least squares (curvilinear)	curve	all points influential
most	least squares (linear)	straight line	all points influential